AN ABSTRACT OF THE THESIS OF

Funfay Jen for the degree of Master of Science in Computer Science presented on February 18, 2020.

Title:

An Analysis of Students' Self-efficacy and the Effectiveness of Peer Review in First-year CS Courses

Abstract approved: _____

Jennifer Parham-Mocello

Students' success is one of the foremost objectives in higher education, and their self-efficacy plays a prominent role in students' achieving their full potentials. It is especially important in STEM fields, which often suffer from higher attrition rates. Therefore, it is important to understand students' self-efficacy levels at an early stage in an effort to retain students and enhance diversity. In this study, we look at the self-efficacy of students participating in a first-year introductory CS class and analyze the trends over the course. We also study the demographic differences across gender and class standing with regard to self-efficacy. Lastly, in a succession of research questions, we ask and answer the question of whether the adoption of a new platform to streamline the students' experience with peer reviews on assignments in the course is justified. We do not find a difference in self-efficacy among students of different gender. However, we do find that freshmen students begin with lower self-efficacy than sophomore students, and the self-efficacy of freshmen students consistently improves after each peer review. While peer review in first-year courses may improve the self-efficacy of some students, the data from the use of a new peer-review platform in this study suggests that more research needs to be conducted before using the tool's automated grading system in a first-year computer science course.

An Analysis of Students' Self-efficacy and the Effectiveness of Peer Review in First-year CS Courses

by
Funfay Jen

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented February 18, 2020
Commencement June 2020

Master of Science thesis of Funfay Jen presented on February 18, 2020

APPROVED:

Major Professor, representing Computer Science

Head of the School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Funfay Jen, Author

ACKNOWLEDGEMENTS

I am indebted to many people who had shaped and continue to shape my academic experience. I am deeply grateful to the important people who came to my help in times that I needed it most. My academic journey was not a smooth one and I was privileged to get to interact with amazing people who appeared at various points along the road and who shared the ride with me.

I would like to express my deep gratitude towards my research supervisor, Dr. Jennifer Parham-Mocello, during my master's degree in computer science for her advisory. It was under her guidance that I got to appreciate what research was like, the details it entailed, and the open-endedness characteristic of any research endeavor. I got a glimpse of what has been done and realized the amount of inquiry yet to be carried out in CS education in particular. It was during many of those personal meetings with her that I reaffirmed my ideas as I established my research questions, learned how to angle them in order to shed the most light on questions of educational interest and sought to find answers with potential curriculum design implications. Through it all, Dr. Parham-Mocello provided lots of insights on education and curriculum design and introduced me to the ins and outs of the logistics. Her passion for the field was always a great inspiration for my work. A professor, mentor and friend, she is always the most understanding and supportive. She makes sure that you are making progress, but also understands what it takes for things to come into fruition. I will be forever grateful for her mentorship so that I could finish my thesis as well as complete my coursework.

I would like to thank all of my committee members for spending time with me and providing valuable feedback. In particular, I'd like to thank Dr. Claudio Fuentes for meeting with me and reviewing the statistical methods employed in the data analysis, as well as giving me early feedback on my work. This gave me some reassurance on the validity of the statistical methods used and whether I was making sound inference and sensible conclusions, and led me to think more about the limitations of my research. I'd like to thank Dr. Prasad Tadepalli, who gave me some initial encouragement on pursuing an MS degree, for agreeing to be on my committee and helping me with various administrative tasks as a general advisor. I'd like to thank Dr. Jason Fick for meeting with me and learning about my research. His

encouragement and interest in my research gave me further motivation to keep on the good work.

I wish to thank Nicole Thompson, the graduate program coordinator during my brief time as a student in the ECE department, who helped me in transitioning to and familiarizing with the department. I wish to thank Calvin Hughes, the incumbent coordinator for the School of EECS, for his kind help during my graduate study in the CS department. Both persons were incredible source of wisdom in matters regarding EECS and graduate study in general, and wonderful people to get to know.

TABLE OF CONTENTS

LIST OF TABLES

An Analysis of Students' Self-efficacy and the Effectiveness of Peer Review in First-year CS Courses

1 Introduction

Students' self-efficacy in their academic career is an important aspect in their overall success and sense of accomplishment. It is even more critical for the success of students majoring in science, technology, engineering, and mathematics (STEM) fields, which traditionally suffer from low enrollment and high drop-out rates. In fact, only a couple years ago, the trend at OSU was that the DWF (drop-withdraw-fail) rate for the introductory computer science class (CS 160) was around 20%. This rate has improved the past two years: Fall 2017 saw a decrease in DWF rates to 7-9% and Fall 2018 it was 9-11%. One possible explanation is smaller class size, since the class was split into 3 100-person sections. However, even a 10% DWF rate is worrisome in an orientation class. The nature of computer programming requires a lot of practice and discipline, and first-year students often find the introductory classes challenging. Sometimes they might feel overwhelmed by the amount of new material and underestimate their true potential, and sometimes the other way around. This aspect of self-efficacy is especially important for female students, as the instructors for these classes often find that women tend to underestimate their achievements in the course. For this reason, the first-year CS classes at Oregon State University integrated peer review into the curriculum. This allows students to receive peer feedback on their assignments and compare what they are doing with others in the class. Primarily, peer review is given on the design part of the program, but sometimes, it is given on the code itself or both. Starting Fall 2018, instructors also introduced Peerceptiv, an online peer reviewing platform, to some first-year intro-level CS classes based on favorable recommendations from OSU faculty in other departments. With the more sophisticated peer reviewing platform compared to what was being used in past years, instructors expected that more guide would be provided to the students and better insight would be gained into the their engagement levels. After all, the hope is that through the peer interaction, students gain valuable insights into different

ways of solving a problem, good coding habits and other useful comments on their work, thereby enhancing the accuracy of self-efficacy. The research in this paper investigates the effectiveness of Peerceptiv and the fairness regarding incorporating it as part of the overall course grade by analyzing the course grades, student responses and attitudes, and demographic differences.

2 Background

The ever growing need for adequately trained professionals in science, technology, engineering, and mathematics (STEM) fields has been met with a decline in students in these majors, a problem known as the STEM attrition [1, 2]. As a STEM field, the computer science major is no exception [3, 8], and freshman interest in CS is on the decline [4]. Although much of the evidence for high dropout rates and failure rates in CS majors is anecdotal, it is common perception that many CS programs are not successful in attracting and retaining potential CS students, nor in helping them discover their self-values for the major. Furthermore, it is observed that women in computing are served even more poorly than their men counterparts [3], with retention figures declining even more rapidly [5, 6]. One commonly raised reason for this is that female students tend to be more interested in the applications of technology, as opposed to computing for its own sake [3, 6]. The gender imbalance of the CS majors also creates its own problems, such as failure to have a more diverse student body with more voice from the female students. Many universities are actively working to counter these challenges while acknowledging that there is more work to do than that has been done.

One area of CS education that has come to researchers' interest is the self-efficacy of CS students. In his book, Bandura [7] defined self-efficacy as "beliefs in one's capabilities to organize and execute the courses of action required to produce given attainments". In general, research has demonstrated that self-belief has influence on academic achievement [9], and there is also research on programming showing that self-efficacy has an effect on course performance and success in certain programming task [10, 11]. Therefore, the question of whether and to what extent students possess adequate self-efficacy is of crucial interest as we assess their success, and more specifically, whether the course offerings at OSU serve their purpose. This is especially important for first-year classes because students' success in their first-year experience can often be important factors for continuing in their major, and studies have shown that self-efficacy is closely tied to retention rates [12, 13]. Research also suggests differential experiences in regards to retention rates for men and women [14]. In line with Bandura's definition, this study focuses on a measurable manifestation of self-efficacy and uses this definition to gauge the change in students' self-efficacy per assignment and over the period of the course. It is our hope that by analyzing

and understanding the patterns through self-reported beliefs and using the findings of this study as one of the initial steps of further pedagogical research and reform, we can better serve our first-year students in the future.

Another feature of the CS 16X sequence (equivalent to first-year CS 0-1-2) at Oregon State University is the use of peer reviews. The practice of peer-reviewing finds its way to various settings due to its beneficial outcomes. Peer review is commonly practiced in academia for publications in academic journals that subject papers to close scrutiny and generate constructive feedback. It is widely seen as a way to safeguard the quality standards and improve the quality of the paper in question by having outside experts review the academic work and provide insight into potential problems or possible errors in a paper [15]. The same idea can be extended to classroom settings, which has been implemented in many English as a Second Language (ESL) classes [16, 17, 18, 19, 20]. For L2 writers for example, peer reviews can be an excellent supplement for traditional instructor-based assessment of students' work, because they "enhance a sense of audience, raise learners' awareness of their own strengths and weaknesses, encourage collaborative learning, and foster the ownership of text" [20]. In one study [21], researchers came to the conclusion that "self-assessment was somewhat idiosyncratic and therefore of limited utility as a part of formal assessment. Peer-assessors on the other hand were shown to be internally consistent and their rating patterns were not dependent on their own writing performance, which is a potential reason why peer assessment can help better gauge the students' performance and give them useful feedback too. Instructors can use the results from the reviews to better understand how the students are doing and whether the instruction is effective. When it comes to computer science, peer review also has many potential arenas [22], although there is relatively less research that has been done regarding its use in CS classes. Among the studies that investigate the practice of peer reviewing, mixed results are reported [23, 24, 25, 26], confirming some of the known benefits of peer reviews and also acknowledging some limitations, such as the need for external motivation. In addition, other researchers studying peer reviews in introductory classes reported similar positive findings [21, 27], and they showed that they can be as accurate as accredited evaluation standards and yield better course performance. Perhaps a little surprisingly, not only are there tremendous benefits on the students who receive the peer feedback of their own work, research shows that the students who give meaningful peer reviews benefit even more than the students

who receive them [16]. This effect manifests more strongly in students at lower skill levels than students at a higher skill levels in a course. Since the students taking the first-year CS sequence come from diverse backgrounds and different majors, it is interesting to see whether students of different class standings benefit differently from the process of peer review. We think that peer review activities are essentially a form of active learning, as opposed to traditional learning where students only learn from lecturing. In a comprehensive metaanalysis [28] on active learning in STEM majors, researchers collated information from 225 studies and reached their conclusion that active learning can help us arrive at the goal of increasing students scores and lowering failure rates.

There can be different modes of peer reviews [17]. In this class, what we are adopting is asynchronous CMC (computer-mediated communication) via a platform called Peerceptiv. The choice of using online peer review is motivated by its convenience, and the difficulty in implementing other modes of peer review. There have also been some recent innovations on implementing in-flow peer review [31], but it is more suited for larger-scale assignments. Our focus is on first-year, lower-division CS courses. There are other implementations of peer review systems, such as SWoRD, that were traditionally developed for writing courses [29, 30] and also programming-specific ones[32, 33, 34]. To the best knowledge of the researchers, Peerceptiv is among the most popular peer-review platforms, and is used in various courses by many institutions. In fact, according its originator's website [35], "SWoRD is now commercially distributed under the name of Peerceptiv" and is continually under research and development. We anecdotally heard from Peerceptiv that its use and effectiveness for programming/CS classes still requires more research due to limited data. Since we decided to adopt it as the peer review platform for CS 160 in Fall 2018 and CS 161 in Winter 2019, it is of particular interest to see whether it's worthy to continue using it in the future. Nevertheless, the Peerceptiv platform is in general a tried and true, "research validated peer assessment tool" [36]. It's the result of a decade of research development focused on the algorithms that evaluate the students review accuracy and other peer-review-related scores. So we use these data as relatively objective metrics to compare with students' subjective self-efficacy scores and attitudes after peer review, and it is of interest to see whether they correlate well. Furthermore, since Peerceptiv holds students accountable by assigning them grades commensurate with their computed scores received for their peer reviews, we are in-

terested in knowing whether it's fair to incorporate the peer-review grades as part of the overall course grade. For the instructors at OSU, it's the first time Peeceptiv is integrated in the CS 16X curricula as hopefully a valuable method for engaging students in assignments more complex than multiple-choice questions and for obtaining an insightful peek into how students are doing. Coupled with surveys given throughout the course, we seek to assess the usefulness and fairness in the peer-reviewing process and its grade evaluation.

There are four general research questions our study seeks to answer, and we carry out the analyses as follows.

RQ 1: Do students' self-efficacy levels improve over the course?

RQ 2: Does the change in self-efficacy diverge along demographic lines?

RQ 3: Is higher level of engagement in peer review as measured by Peerceptiv associated with more enhanced self-efficacy among students?

RQ 4: Do the objective metrics evaluated by Peerceptiv correlate well with students' subjective attitudes toward peer review?

RQ 5: Do the peer review grades correlate well with the overall grades in the course?

3 Methods

3.1 Class Structure

Oregon State University structures its first-year computer science courses in a three-quarter series, CS 160-1-2. It's worth noting that CS 160 is not a prerequisite for CS 161 and CS 162, and it serves as a gentle orientation to the field of computer science. Therefore CS 160 has some more higher-level concepts than its sequel. Since CS 160 is not a prerequisite, there is also some amount of overlap in material between CS 160 and CS 161, although the programming languages are different. CS 160 uses Python and CS 161-2 use C++. In CS 160, there are 5 assignments that are composed of a mixture of design/programming problems (examples include making a calculator, summing/integrating a function, Hangman game, etc.) and general questions geared towards orientation (examples include making a plan for the near future in CS). CS 161 goes deeper into problem solving and covers in detail expressions and statements, flow of control, functions, pointers, error handling and debugging. In CS 161, there are six assignments (examples include Connect Four, producing fractals, etc.) that emphasize the computational aspects, as well as some fundamental programming constructs using C++ as the lingua franca. The difficulty of the assignments are significantly increased compared to those in CS 160, but much more manageable in size compared with those in CS 162, which has a focus on Object-Oriented Programming (OOP) with more complex programs both in size and in logic.

In all of the first-year CS classes, the lectures typically consist of presenting slides on conceptual programming concepts and annotating them as the instructor expands on certain topics with live coding periods in which the instructor demonstrates some examples through a terminal. Assignments given throughout the quarter aim to reinforce the concepts via the design and programming of various tasks. In addition to regular assignments, students are also required to participate in peer-reviewing work turned in by classmates on Canvas. While the first-year courses are required for pro-school admittance, some students skip some of them because they have transferred credits from elsewhere or have taken high-school advanced classes on the same topic and passed the CS AP test. Due to the growing importance of computing in STEM fields and others, there are also many students who elect to take these classes

out of interest, even though they are not required to. These all contribute to a diverse student composition in the first-year CS classes with students from various majors and differing class standings, which make for an ideal place to experiment with new teaching tools and pedagogical reforms. In fall 2018 and winter 2019, researchers at OSU decided to introduce Peerceptiv, a more sophisticated peer-review platform, to the curricula as a more cohesive way to organize the activity. One of the research goals of this study is to assess the effectiveness of this new tool to decide its use in future curriculum developments.

3.2 Peer Review

This study focuses on the self-efficacy of students in CS 160 (and the same methods can be applied to CS 161, its sequel) as a starting point for understanding students' perceptions in more advanced classes. It is also a tradition that students in these first year programming classes review each others' work as a way to give constructive feedback to and learn from each other. In the past, this process occurred on Canvas, an online platform used by OSU for virtually anything related to a course. It has a comprehensive utility for managing course materials, submitting homework, and grading assignments, among many others. As for peer review, instructors can randomly assign peers to each student, and they can receive comments regarding their submission. However, there are some drawbacks using Canvas as the peer-review platform. Firstly, from the researchers' experience, while Canvas is good for simple assignment management and has some basic features needed for peer-reviewing, it is not specifically designed for peer review in its own right. There is limited flexibility for configuring the ways peer reviews can be administered, and students can only submit text entries for the peer reviews as there are no options for other types of input (e.g. multiple choice). In addition, there is no channel for back-evaluations where students being reviewed can give some comments on the feedback they received from their peers. More importantly, there is no way to measure *how* good the peer review is, as we don't have an effective way to distinguish high quality reviews from low quality ones, nor are they graded according to any quality measure. As a result, we often see students submit some brief comments as an obligatory chore for the sake of completion, which is not fair for those who receive them and students who spent time and gave thoughtful reviews.

Prior to this study, Peerceptiv, as a candidate for more effective peer-review management tool, was introduced to the researchers by other faculty members at OSU. Besides having all the extended functionalities mentioned above that Canvas lacks, one of its major benefits is that it offers a way to evaluate the engagement of students in the peer-review process. Figure 3.1 shows a typical interface to the student for an assignment, which allows for some text entry as well as multiple choice entry for some fixed-scale dimensions of the particular assignment, and Figure 3.2 displays the entire set of peer reviews and back-evaluations for a student on one assignment:

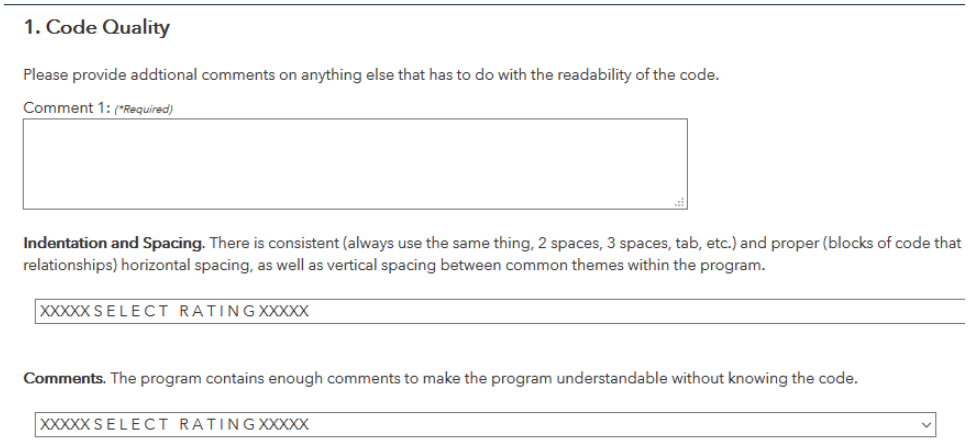


FIGURE 3.1: Screenshot of the peer review interface in Peerceptiv

After students have submitted the reviews, it computes a number of scores associated with the peer reviews, including review accuracy and review helpfulness. According to Peerceptiv, there are 3 components of the Overall grade in Peerceptiv:

- a Writing grade — the grade received on the assignment submission by the peers and by the instructor (if the instructor wishes to participate in the assessment.) These may be written submissions or any assignment upload requiring formative feedback.
- a Reviewing grade — a measure of the accuracy of review ratings and helpfulness of review comments.
- a Task grade — a completion grade for all reviewing and back-evaluation tasks.

Of those, the key scores that Peerceptiv provides that offer additional insight compared to other basic peer review systems, such as that built in Canvas, are the accuracy and helpfulness scores, which comprise the reviewing grade for the student.

- Accuracy grades measure how closely the ranking order of the ratings a reviewer provides on peer documents corresponds to the ranking order of peer rating averages for each rating prompt on those same documents. When students

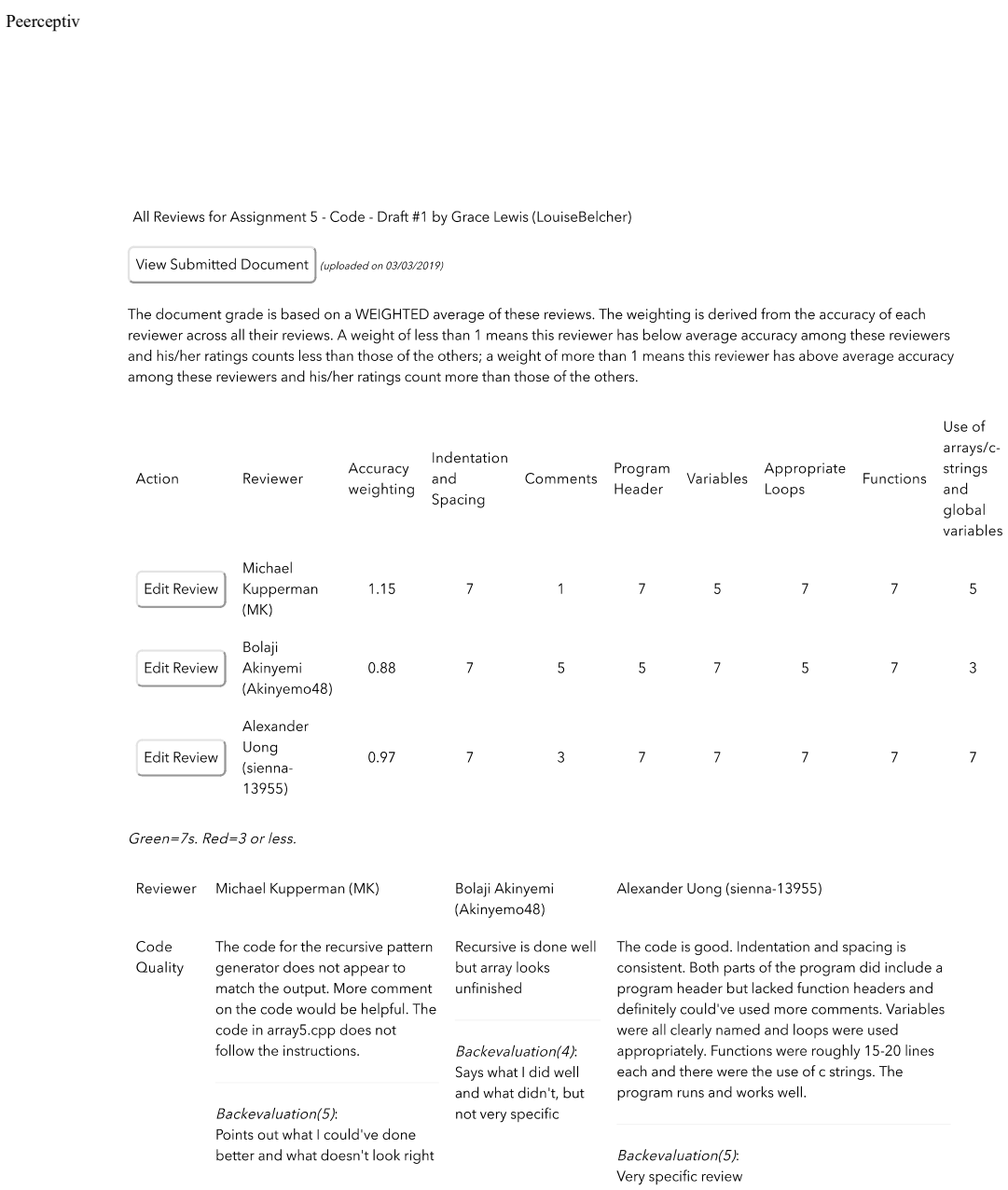
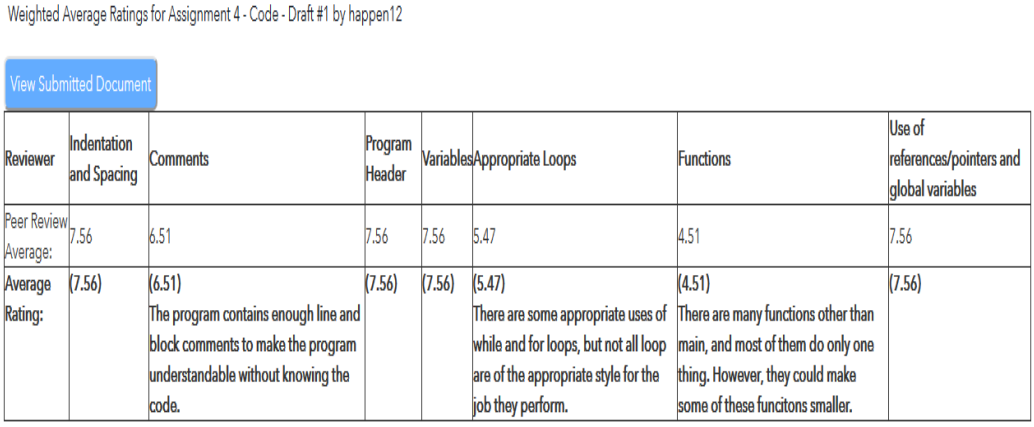


FIGURE 3.2: Screenshot of the peer reviews including back-evaluations

provide ratings that track with the relative order of mean ratings for those same reviewing dimensions on the same documents, they are rewarded with a higher accuracy score. [The system calculates the Spearman correlation across the number of papers reviewed (n) times the number of rating prompts (m).] If a reviewer gives the same rating to all papers on all dimensions (e.g., 7s across the board), this necessarily results in an accuracy score of 0. Negative correlations are theoretically possible (i.e., an opposite view from everyone else) but rarely occur. Accuracy scores (-1 to 1) are then curved to produce accuracy grades (0 to 100) using the grading curve settings.

- The helpfulness grade is the extent to which authors believed a reviewer's comments were helpful and specific, as awarded by the back-evaluation ratings (5 high to 1 low.) The first step is to normalize each author's ratings (some authors might be especially harsh or overly nice across the board), and the final step is to curve the Helpfulness scores according to the instructor's settings.

The complete specification provided by Peerceptiv is given in appendix .1. Figure 3.3 shows a screenshot for the average ratings a student received on an assignment.



Weighted Average Ratings for Assignment 4 - Code - Draft #1 by happen12

View Submitted Document

Reviewer	Indentation and Spacing	Comments	Program Header	Variables	Appropriate Loops	Functions	Use of references/pointers and global variables
Peer Review Average:	7.56	6.51	7.56	7.56	5.47	4.51	7.56
Average Rating:	(7.56)	(6.51) The program contains enough line and block comments to make the program understandable without knowing the code.	(7.56)	(7.56)	(5.47) There are some appropriate uses of while and for loops, but not all loop are of the appropriate style for the job they perform.	(4.51) There are many functions other than main, and most of them do only one thing. However, they could make some of these funcitons smaller.	(7.56)

FIGURE 3.3: Screenshot of average ratings on various dimensions a student received on an assignment

Based on the grading algorithm, Peercpetiv tallies up the various scores and gives an overall peer review grade to each student, allowing the instructor to incorporate it in the overall course grade. Figure 3.4 shows a screenshot of what the students can see for his/her assignment grade.

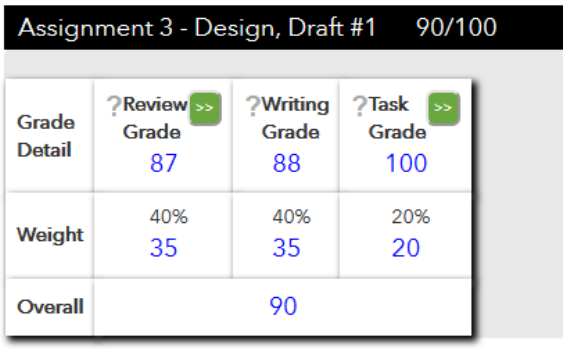


FIGURE 3.4: Screenshot of the assignment grade in student view

This can be a useful feature not only for the instructors who can now have some insight into how the class is doing, but also it can be motivating for the students to engage proactively in the reviewing process, as now they have something holding them accountable. Better accuracy, helpfulness and completion is rewarded with high grades, and slackers are punished by getting lower grades. It was the researchers' hope that by using a more sophisticated peer-review platform, we could foster a higher level of engagement in communal collaboration and critique and bring out the most peer reviews can offer.

Meanwhile, since Peerceptiv is more extensively used in other disciplines and for the first time adopted in the first-year CS classes, the use in CS is experimental by nature, and the researchers are interested in evaluating its effectiveness and suitability for wider and continued use. Currently, it has only been used in CS 160 and CS 161 in which the assignments are more manageable in size compared to CS 162 and other advanced classes. For example, CS 162 makes heavy use of OOP which often requires separate compilation with multiple files. It would be more difficult to administer peer reviews involving multiple files, both logistically and technologically.

3.3 Procedures and Participants

The researchers completed the CITI training and obtained IRB approval prior to the start of the study. Surveys are given to students to collect general background information of the class, as well as to obtain peer-review-specific information about the students. There are three different surveys: a pre-survey administered at the beginning of the course (see appendix .4 for relevant parts), a pre-peer review survey administered before each of the assignments, and a post-peer review survey admin-

istered after each of the assignments. A brief display of the key questions are shown below (see appendix .2 and appendix .3 for a complete description of the peer review survey questions which include questions not used in this study that are intented for other research projects):

Pre-survey: This is a survey that gathers some basic background information such as programming experience and attitudes, as well as asks for consent from students to be included in the study. Relevant questions are whether the student agrees to storing their responses for future studies and whether they agree to participate in this study.

Pre-peer review survey: This is a survey given prior to the peer review as a predictive activity. Students are asked about their perception of the score they think they can get after working on some tasks. Some auxiliary information is also collected. After this survey they will take part in the actual peer review.

Post-peer review survey: This is a survey given as a reflective activity following the peer review. Students are asked the same information requested in the pre-peer review survey, with the addition of questions regarding their attitudes towards the peer review activity and whether they think it's useful.

In CS 160, students work on some tasks before taking the predictive survey, and then they work on the peer reviews. After that, they are given the reflective survey. This cycle repeats for each of the assignments throughout the quarter. Compared to CS 160, there is some major improvement in CS 161 in terms of the execution of the plan: the predictive and reflective surveys are accessible to students in separate assignments under different timeframes, enforcing a stricter separation of the two activities and allowing for true reflection after each peer review. This is in contrast to what was done in CS 160, where the predictive and reflective activities are accessible in the same assignment. Although it is expected that the students finish them at different times with the peer review activity in between, it is theoretically possible that the students can carelessly submit the two responses in one go, undermining the validity of the data.

Table 3.1 shows the number of consenting participants in CS 160 and the demographic information for these students.

	CS 160
Consent Rate	73.4
Female	33
Male	83
Major	102
Non-major	14
Freshman	59
Sophomore	41
Junior	10
Senior	6

TABLE 3.1: Demographics of consenting participants

4 Results and Discussion

4.1 Improvement in Self-efficacy

RQ 1: Do students' self-efficacy levels improve over the course?

Students participated in peer reviews for each of the assignments in the course and were given surveys, once before the peer review (which was after they submitted some tasks to be peer-reviewed and before they started working on the next task) and once after, requesting their prediction for what scores they would get for each assignment. We are interested in learning whether their post-peer review self-predicted scores are closer to their actual scores received after the peer review, and whether this trend improves as the course progresses. This is of interest because we want to know whether having their work peer-reviewed and reviewing work of others enhance the accuracy of students' self-efficacy, as students get exposed to and become more experienced with this form of active learning, and thus potentially improve retention rates for students majoring in computer science.

The relevant variables collected are:

- *pre*: the self-predicted score for what the student thought they would get for a particular assignment, before they started working on it and having it peer-reviewed.
- *post*: the self-predicted score for what the student thought they would get for a particular assignment, after they turned in the assignment and the peer review.
- *actual*: the actual score the student received, graded by the TAs.

We have those variables for each student and for each of the assignments.

This is a multivariate data set, and the suitable statistical method is Hotelling's T^2. To measure the students' self-efficacy level in the spirit of [7], this study defines self-efficacy in terms of the absolute difference between the students predicted score on an assignment and the actual score they received. With this definition, we will perform t-tests on the improvement in the gap defined as:

$$|pre - actual| - |post - actual|$$

for earlier assignments and for later assignments, accounting for the correlation using Hotelling's T^2. The more the gap "closes" (positive number), the more self-efficacy the student has gained and vice versa.

There were 5 assignments in CS 160 and a total of 13 students who participated in all the 5 sets of surveys for each assignment, so the analysis is performed on a sample size of 13 (as opposed to the class size which was 100+). Assignments are grouped into "earlier" (Assignments 1 and 2) and "later" (Assignments 3, 4, and 5) and averages are computed for earlier and later assignments, their correlation calculated, and the Hotelling's T^2 statistic and confidence intervals are calculated according to the appropriate formulas. The joint confidence intervals would tell us how self-efficacy levels have changed over different periods, after accounting for multiple comparisons.

The test shows no evidence that the 13 students had any improvement in self-efficacy on either the earlier or later assignments (p=0.335), and a 95% joint confidence interval is [-2.38, 8.76] points of improvement for earlier assignments and [-4.66, 2.38] points for later assignments (out of 100 points).

Even though we have a null result for the improvement in students' self-efficacy from earlier to later assignments, we note that the students might represent a unique subpopulation distinct from the majority of the class. After all, only less than 10% of the students did not miss a survey. In fact, a closer look at the data reveals that the 13 students had relatively high self-expectations before and after peer review, and both times they substantially overestimated their true performances. It could be that these students clicked through the surveys quickly just to earn points, or it might be that the TAs graded them inaccurately. Also possible is that the students who took time to participate in all peer review activities had very high expectations of themselves and were not easily moved by the peer reviews. In any case, this was an intriguing phenomenon that might be worth investigating.

A coarser analysis is also performed where we included more students who did not complete all the surveys. A similar analysis on earlier and later assignments can be performed, although they now involve different students. Separate confidence intervals are used in this case and multiple comparisons are not accounted for. This can give us some preliminary insight about whether the class as a whole is somewhat different from the 13 students who completed all the surveys. A t-test on the earlier assignments including all students who supplied response gives a p-value of 0.05066, and a t-test on the later assignments including all students who supplied response gives a p-value of 0.9802. This offers some suggestive evidence at a 95% confidence level that the class as a whole might have improved and stabilized their self-efficacy

as the course progressed. However, further study needs to be done to verify that.

To summarize, we have some preliminary conclusions that participating in peer reviews is not associated with a statistically significant change in a small group of students' self-efficacy in the quarter. Whether this conclusion extends to the whole class is questionable. In fact, we are seeing some suggestive evidence that the class has improved after two peer review activities.

On the other hand, there's some threats to validity in our design and analysis of the study. Specifically, it is not clear whether the self-predictions and the scores received were accurate. It is possible that some students might click through the surveys just to get credit, thereby compromising the validity of the self-reported data. Also, the graders might not have given the most accurate scores to the students. That was because the particular assignments in the intro-level CS class involved the *design* of programs, which involved some subjectivity when it came to grading.

4.2 Demographic Differences

RQ 2: Does the change in self-efficacy diverge along demographic lines?

For this research question, the assignments are analyzed separately for the students' self-efficacy in terms of the closing or widening of the gap between predicted and actual received scores, and demographic differences are detected by appropriate tests. To compare gender differences, t-tests can be used to compare differences in self-efficacy between male and female students. There are different ways to compare students in various majors and class standings. One method is to use one-way ANOVA for an initial screening of any differences among students from different majors and class standings. If there is evidence for such differences, the Tukey-Kramer procedure is employed to make simultaneous comparisons to obtain adjusted confidence intervals and p-values. An alternative way is to only compare freshmen and sophomores. We mostly use the second approach but also present some results from the first. The findings follow.

None of the t-tests on gender differences yields significant results. Table 4.1 sums up the p-values and low and high ends of 95% confidence intervals for the difference in self-efficacy (measured by the "gap" in number of points out of 100) between genders for the 5 assignments (Assign 1 – Assign 5).

The class standings of participants in CS 160 mostly consist of Freshmen and

	p-value	CI
Assign 1	0.82	(-2.76, 3.46)
Assign 2	0.88	(-3.08, 3.58)
Assign 3	0.17	(-4.63, 0.82)
Assign 4	0.13	(-1.00, 7.81)
Assign 5	0.11	(-0.44, 4.23)

TABLE 4.1: Gender differences in self-efficacy in CS 160

Sophomores, along with some Juniors but very few Seniors. The initial one-way ANOVA analysis excludes the seniors, and found that out of the 5 assignments, 2 of them have significant p-values (Assign 1, p = 0.0249 and Assign 4, p = 0.00449). In addition to confirming this finding, The Tukey-Kramer procedure identifies that the sophomores have lower mean values than the other groups, indicating that they gained less in the peer review. An example using data from assignment 4 is given in Table 4.2.

Linear Hypotheses	Estimate	Std. Error	t value	Pr(>|t|)
Junior - Freshman == 0	0.55	5.46	0.10	0.99
Sophomore - Freshman == 0	-9.47	3.00	-3.16	0.005 **
Sophomore - Junior == 0	-10.02	5.34	-1.87	0.14

TABLE 4.2: Differences in self-efficacy among class standings in CS 160

However, in our case, we don't have many juniors in the class and some of them did not complete the assignments or take the surveys, making the sample size for juniors quite small. Therefore, a better approach draws focus on the freshmen and sophomores only. Standard t-tests can be used for this purpose. Table 4.3 summarizes the p-values and low and high ends of 95% confidence intervals for the difference in gain in self-efficacy (measured by the "gap" in number of points out of 100) between freshmen and sophomores for the 5 assignments (Assign 1 – Assign 5), as well as the freshman and sophomore means in the last two columns.

Although we seem to get mixed results across different assignments, we do see two strongly significant results (Assign 1 and Assign 4), which is consistent with the one-way ANOVA findings. In addition, it is interesting to observe that the point estimates for the freshmen's gain in self-efficacy are consistently larger than those for

	p-value	CI	Freshman mean	Sophomore mean
Assign 1	0.017	(0.85, 8.45)	2.33	-2.32
Assign 2	0.21	(-1.14, 5.13)	0.61	-1.38
Assign 3	0.94	(-2.36, 2.55)	-0.76	-0.85
Assign 4	0.003	(3.21, 15.73)	3.87	-5.60
Assign 5	0.35	(-1.60, 4.53)	0.10	-1.36

TABLE 4.3: Differences in gain in self-efficacy between Freshmen and Sophomores in CS 160

the sophomores' gain in self-efficacy. Consulting the grading scale for CS 160 (Table 4.4), they represent at least half a letter grade, if not much more.

Grade	Average
A	93 or greater
A-	90 - 92
B+	87 - 89
B	83 - 86
B-	80 - 82
C+	77 - 79
C	73 - 76
C-	70 - 72
D+	67 - 69
D	63 - 66
D-	60 - 62
F	less than 60

TABLE 4.4: Grading scale in CS 160

This suggests that the peer review activities might indeed have a higher impact on the freshmen than on the sophomores, and the reason we don't have significant p-values across the board might be due to the lack of power of the tests, which in turn might have to do with excessive variations due to confounding variables. For more context, we also tabulate the absolute difference between self-predicted scores and actual received scores *prior to* the peer reviews for freshmen and sophomores in Table 4.5

Once again, we observe that freshmen consistently have higher mean values

	Freshman mean	Sophomore mean
Assign 1	16.82	9.20
Assign 2	8.43	6.11
Assign 3	9.29	9.09
Assign 4	11.42	7.33
Assign 5	12.11	5.34

TABLE 4.5: Self-efficacy prior to peer reviews for Freshmen and Sophomores in CS 160

than sophomores. In this case, this represents the distance of the predicted score from the actual score prior to peer reviews; that is, their self-efficacy prior to peer reviews (smaller values indicate higher self-efficacy). In conjunction with Table 4.3, we hypothesize that the freshmen start off with lower self-efficacy than their sophomore counterparts, but also improve through the peer review activities much more than the sophomores do. All of the above hypotheses are still tentative as we have not observed statistically significant p-values across the assignments, so future studies should focus on variation reduction (such as by having finer grading rubric and giving partial points) and more control over the confounding variables.

Lastly, as for any differences among various majors, it turns out that in Fall 2018, students who took CS 160 were predominantly in Pre-Computer Science, so we defer any analysis to future studies that have a more diverse student body.

4.3 Effectiveness and Fairness of the Use of Peerceptiv

One question this study set out to answer was whether continued use of Peerceptiv is justified. The research questions we have answered so far does not explicitly factor in students' engagement in Peerceptiv. The following research questions are designed to provide some insight as to the usefulness of Peerceptiv activities in regard to students' self-efficacy, attitudes, and overall grades in the course.

RQ 3: Is higher level of engagement in peer review as measured by Peerceptiv associated with more enhanced self-efficacy among students?

To answer this question, we did some exploratory data analysis to corroborate our hypothesized model of students' self-efficacy in regards to its manifestation in the variables measured in concrete numerical terms. Specifically, we plot the student's

predicted score after the peer review against the predicted score before the peer review, as well as the actual score and the overall Peerceptiv grade they received on that assignment. Some sample exploratory plots are given in Figure 4.1,

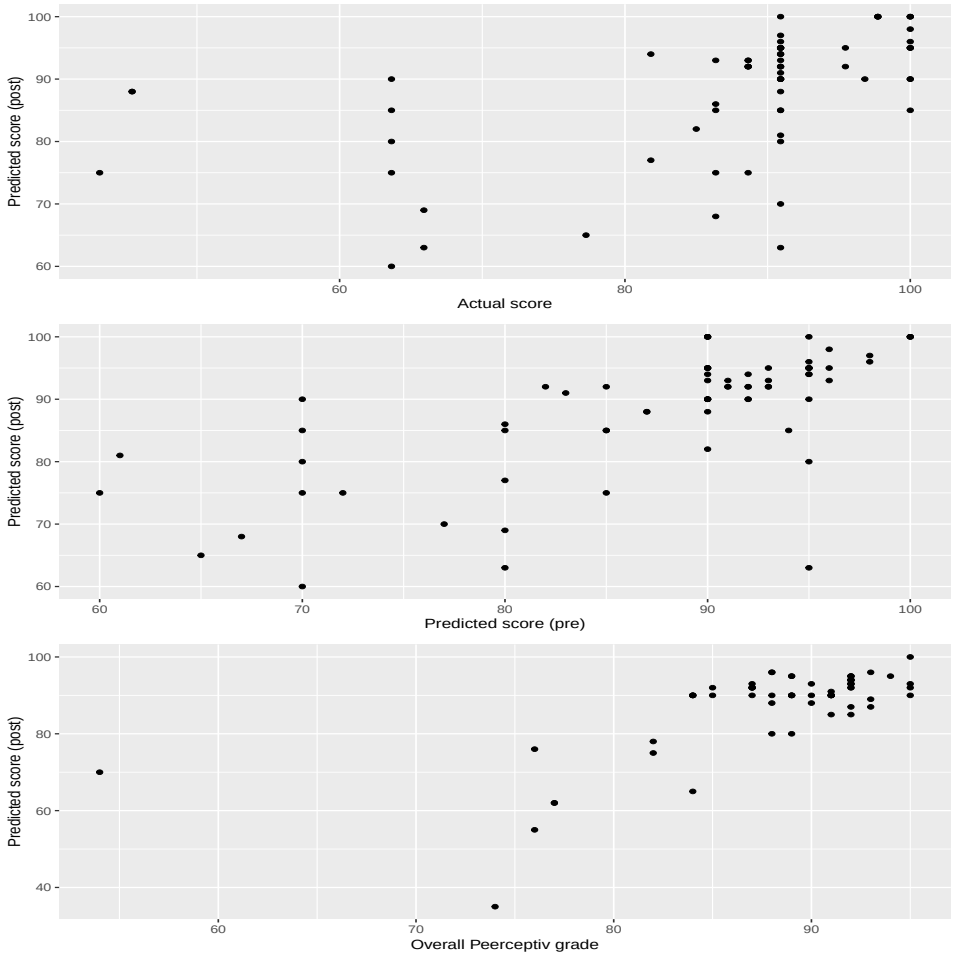


FIGURE 4.1: Sample exploratory plots depicting the relations between predicted scores, actual score and the overall Peerceptiv grade

We observed relatively consistent linear trends among the variables of interest. Furthermore, we are interested in understanding any possible interactions between some of the variables, which can in turn help answer our research questions. In light of these observations, we use the following linear regression model for students' predicted score on an assignment after the peer review in terms of their predicted score on that assignment before the peer review, their actual received score, and the overall Peerceptiv grade they receive for their effort and quality in the peer review activity.

$$\begin{aligned} \mathrm{E}(post \mid pre,\ actual,\ overall) &= \beta_0 + \beta_1 actual + \beta_2 pre + \beta_3 overall \\ &\quad + \beta_4 (actual \times overall) \\ &\quad + \beta_5 (pre \times overall) \end{aligned}$$

where *pre*, *post* and *actual* are the predicted score before the peer review, predicted score after the peer review, and the actual received score respectively, as previously defined. The additional variable *overall* denotes the overall Peerceptiv grade as computed by algorithms in Peerceptiv. The interesting terms in the model are the interactions, whose presence would indicate the effect of the overall Peerceptiv grade on the self-efficacy of the student on the assignment. Ideally, if a higher overall Peerceptiv grade for peer review is associated with improved self-efficacy, then β_4 should be positive and β_5 should be negative. In other words, higher marks achieved on Peerceptiv should make one's prediction more based on their actual score and less on the pre-peer review prediction. Whether that's indeed the case is the subject of the ensuing analysis. We again analyze the assignments separately and observe the p-values and effect sizes of the interactions terms. No multiple comparison issues are heeded here.

Table 4.6 summarizes the point estimates and their p-values of the relevant coefficients for 5 assignments in the quarter. Due to some dubious data points (most likely due to invalid self-predicted scores provided by students), the linear regression diagnostics show some problems with fitting the entire data set. After restricting the analysis to students who mostly predicted themselves as "passing" (above C for CS majors at OSU), the regression assumptions are met. The interesting coefficients here are β_4 and β_5, and the values for β_1 and β_2 are listed to compare effect sizes. There is no clear indication of interactions. The p-values for the interactions are significant for some assignments but not for others. Furthermore, the signs for the interactions are not consistently compatible with the postulated hypothesis, which would predict that β_4 is positive and β_5 is negative. The practical significance of the point estimates are not great either.

The overall Peerceptiv grade consists of a comprehensive set of measures for the level of engagement in peer reviews and includes some extraneous factors in grading. Would it be possible that metrics more intimately tied to the peer-reviewing activities

are more indicative of any improvement in self-efficacy? Accuracy in peer reviews is one such metric as it requires the students to understand each others' work and provide constructive feedback. To help further answer this question, we also analyzed the same data using the average accuracy scores of the students in their peer reviews, and the results are similar.

In summary, there does not seem to be a strong relationship between the Peerceptiv grades and the improvement in self-efficacy across the 5 assignments, at least for students who predicted themselves as passing or above.

	Assign 1		Assign 2		Assign 3		Assign 4		Assign 5	
	Est.	p-val	Est.	p-val	Est.	p-val	Est.	p-val	Est.	p-val
β_1	-0.11	0.634	-0.076	0.66	1.31	0.015*	-1.058	0.13	0.70	0.19
β_2	4.80	5.9e-5*	0.10	0.63	0.045	0.0002*	2.68	0.0015*	0.17	0.76
β_4	2.4e-03	0.42	0.003	0.32	-0.015	0.018*	0.0146	0.072	-0.008	0.20
β_5	-4.8e-02	6e-3*	0.0046	0.20	0.045	5.2e-05*	-0.028	0.002*	0.003	0.61

TABLE 4.6: Coefficient estimates of the self-efficacy model in CS 160

RQ 4: Do the objective metrics evaluated by Peerceptiv correlate well with students' subjective attitudes toward peer review?

To answer this question, we asked students in surveys after each peer review activity whether they think it's useful reviewing other students' work as well as whether they think it's useful receiving other students' feedback. The original survey answers have 5 levels (from not at all useful to extremely useful), but for the data analysis we collapse the 5 levels into 2 so that there are enough data points in each of the two levels. We use t-tests to compare the overall Peerceptiv grades between students who find it useful reviewing their classmates, as well as the accuracy scores in Peerceptiv between students who find it useful receiving reviews from their classmates, all of which is done for each assignment in the class. The reason for the two pairs of comparisons is that the researchers believe that for the use of Peeceptiv grades to be justified, ideally the two pairs of variables should have relatively high correlation. See Table 4.7 and Table 4.8.

There are still some mixed results in these two tables. Although we can't make a conclusive statement, there is some suggestive evidence that the students who find peer-reviews "not useful" achieve higher accuracy in rating their peers, as well as a higher overall Peerceptiv grade. Note that for all but one assignment in either the

	p-value	CI	Not Useful	Useful
Assign 1	0.004	(1.27, 6.22)	90.72	86.98
Assign 2	0.008	(3.83, 24.57)	82.59	68.39
Assign 3	0.69	(-5.22, 3.50)	87.55	88.40
Assign 4	0.02	(0.57, 7.48)	86.63	82.60
Assign 5	0.17	(-11.59, 2.06)	77.04	81.81

TABLE 4.7: Differences in overall Peerceptiv grades between students who find reviewing peers useful and who don't

	p-value	CI	Not Useful	Useful
Assign 1	0.0012	(4.65, 18.12)	86.11	74.72
Assign 2	0.18	(-5.03, 26.12)	74.00	63.45
Assign 3	0.019	(-25.94, -2.48)	71.48	85.70
Assign 4	0.24	(-5.52, 21.87)	66.00	57.83
Assign 5	0.58	(-11.74, 20.87)	54.11	49.54

TABLE 4.8: Differences in accuracy grades in Peerceptiv between students who find receiving peer reviews useful and who don't

accuracy score or the overall Peerceptiv grade, the mean is higher for the students who find the peer review activities not useful than those who do, and the p-values for some of the assignments are significant. This is a rather intriguing phenomenon. One explanation is that those who don't find it useful to review other students' work are better students in other aspects, e.g., they are good at the programming assignments and don't find the need to peer review their work, which might be reviewed by students who are less experienced than them. However, the lack of consistent results and statistical significance on some of the assignments calls for further research. It's highly likely that the noise in the variables is too high to illuminate the true effect. Better control of the conditions should reduce variations, and check confounding variables, and can lead to more consistent outcomes.

RQ 5: Do the peer review grades correlate well with the overall grades in the course?

See Table 4.9.

Lastly, we look at the correlation between the Peerceptiv grade and the course grade. From the table, the correlation coefficients range from 0.12 to 0.55, and thus we don't see a high correlation with the course grade. In conjunction with results

	Correlation with Course Grade
Peer Review Grade 1	0.55
Peer Review Grade 2	0.44
Peer Review Grade 3	0.29
Peer Review Grade 4	0.13
Peer Review Grade 5	0.41

TABLE 4.9: Correlation of the peer review grade with the course grade

in previous research questions, we don't see any demonstrable effectiveness of the Peerceptiv platform for peer reviews. It does not seem to enhance students self-efficacy, and students who earn high marks on peer reviews neither find it useful giving, nor receiving reviews. Moreover, the peer review grade as a measure of engagement in the platform does not correlate well with the overall success in the class, as measured by the course grade. In light of all these points, we suggest that the instructors halt the use of Peerceptiv in future quarters until a more developed version of the software is delivered, especially with some features tailored to computer science classes.

5 Conclusions

The motivation of our study originated from understanding students' self-efficacy in first-year CS classes while keeping an eye on the potential effects from the peer review activities they engage in as part of the curriculum. We first aimed to understand whether there was any improvement in students' self-efficacy after the peer reviews as well as over the course. While we reached a null result and did not find enough evidence to state it positively, there is some suggestive evidence that there might be some improvement in self-efficacy if we look at the data a little differently. Future work should center around controlling for confounding variables in an effort to reduce the variances, as well as to better justify any causal inferential statements. As it stands right now, any conclusion we draw in this study is largely observational.

To further improve the quality of the data collected, future researchers should consider investing more efforts in training the TAs to give more accurate grades, which can entail drafting more refined rubrics to reflect the true quality of student submissions, as well as educating the teaching assistants on what qualifies as good design since it involves some levels of subjectivity and design is heavily used in the intro-level classes. The other possible source of error is the self-reported data from the students collected in surveys. While they are a required component of the course, it was graded based on completion, not accuracy. It is possible that some students clicked through the surveys just to get points for it, compromising the quality of data. It is also possible that some students can have some confusion about the structures of the current surveys as they need to complete multiple questionnaires over the quarter, some of which combined questions from several different research projects. In the future we can split them into smaller and more manageable surveys that can hopefully reduce the cognitive load when taking them.

We also investigated the demographic differences in self-efficacy using background information about the class. First, we made the observation that gender was not implicated in students' self-efficacy, which is reassuring news for our colleagues who work hard on the inclusion of women engineers in our university and beyond. One interesting finding was that freshmen students consistently improved more than their sophomore counterparts after each peer review. Further investigation revealed that the freshmen also started out with lower self-efficacy. These observations align

with our expectation for first-year students: they might have lower confidence due to a variety of factors related to their acclimation to college life. Fortunately, they also improved in self-efficacy very fast, closing the gaps between freshmen and sophomores in some way. Still, due to excessive variances, we were not able to make statistically significant statements about our claims. Future work should focus on controlling for confounding variables and improving data quality, as mentioned above. In the future, if we have a more diverse student composition taking intro-level CS classes, we can also study any difference between people from different majors.

Lastly, in a succession of research questions, we investigated whether the use of a new peer-reviewing platform, Peerceptiv, is justified. The results seemed to be negative. We did not find any strong relationship between Peerceptiv grades and improvement in self-efficacy. Moreover, we asked for students' attitudes towards it and discovered that those who rated it "not useful" actually achieved higher accuracy grades as well as overall grades evaluated by Peerceptiv, an intriguing finding that prompts for further study. As one more piece of counter-evidence, the overall Peerceptiv grades as computed by their algorithms do not correlate well with the overall course grades the students receive. In light of the above findings, we suggest that the instructors halt its use and keep updated on any new developments by Peerceptiv that better target students in computer science majors.

Bibliography

1. Chen, X. (2013, November 26). STEM Attrition: College Students' Paths Into and Out of STEM Fields. Retrieved June 30, 2019, from https://nces.ed.gov/pubsearch/pubsinfo.asp?pubid=2014001rev

2. Hill, S., Slusaw, D., & Wisser, J. (n.d.). Understanding Causes of High Attrition Rates In STEM Majors. 1.

3. Sloan, R. H., & Troy, P. (2008). CS 0.5: A better approach to introductory computer science for majors. ACM SIGCSE Bulletin, 40, 271âĂŞ275. ACM.

4. Vegso, J. (2005). Interest in CS as a major drops among incoming freshmen. Computing Research News, 17(3), 6-1.

5. Camp, T. (1997). The incredible shrinking pipeline. Communications of the ACM, 40(10), 103âĂŞ110.

6. GÃijrer, D., & Camp, T. (2002). An ACM-W Literature Review on Women in Computing. SIGCSE Bull., 34(2), 121âĂŞ127. https://doi.org/10.1145/543812.543844

7. Bandura, A. (1997). Self-efficacy: The exercise of control. Macmillan.

8. Guzdial, M. (2002). Summary: Retention rates in cs vs. institution. Message Posted on Acm Sigcse Moderated Members List, Georgia Tech.

9. Valentine, J. C., DuBois, D. L., & Cooper, H. (2004). The relation between self-beliefs and academic achievement: A meta-analytic review. Educational Psychologist, 39(2), 111-133.

10. Wiedenbeck, S. (2005). Factors affecting the success of non-majors in learning to program. Proceedings of the First International Workshop on Computing Education Research, 13-24. ACM.

11. Ramalingam, V., LaBelle, D., & Wiedenbeck, S. (2004). Self-efficacy and Mental Models in Learning to Program. Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, 171-175. https://doi.org/10.1145/1007996.1008042

12. Devonport, T. J., & Lane, A. M. (2006). RELATIONSHIPS BETWEEN SELF-EFFICACY, COPING AND STUDENT RETENTION [Text]. https://doi.org/info:doi/10.2224/sbp.2006.34.2.127

13. Raelin, J. A., Bailey, M. B., Hamann, J., Pendleton, L. K., Reisberg, R., & Whitman, D. L. (2014). The Gendered Effect of Cooperative Education, Contextual Support, and Self-Efficacy on Undergraduate Retention. Journal of Engineering Education, 103(4), 599-624. https://doi.org/10.1002/jee.20060

14. Sawtelle, V., Brewe, E., & Kramer, L. H. (2012). Exploring the relationship between self-efficacy and retention in introductory physics. Journal of Research in Science Teaching, 49(9), 1096-1121. https://doi.org/10.1002/tea.21050

15. Ware, M. (2008). Peer review: Benefits, perceptions and alternatives. Citeseer.

16. Lundstrom, K., & Baker, W. (2009). To give is better than to receive: The benefits of peer review to the reviewer's own writing. Journal of Second Language Writing, 18(1), 30-43. https://doi.org/10.1016/j.jslw.2008.06.002

17. Chang, C.-F. (2012). Peer Review via Three Modes in an EFL Writing Course. Computers and Composition, 29(1), 63-78. https://doi.org/10.1016/j.compcom.2012.01.001

18. Mangelsdorf, K. (1992). Peer reviews in the ESL composition classroom:What do the students think? ELT Journal, 46(3), 274-284. https://doi.org/10.1093/elt/46.3.274

19. Paulus, T. M. (1999). The effect of peer and teacher feedback on student writing. Journal of Second Language Writing, 8(3), 265-289. https://doi.org/10.1016/S1060-3743(99)80117-9

20. Tsui, A. B. M., & Ng, M. (2000). Do Secondary L2 Writers Benefit from Peer Comments? Journal of Second Language Writing, 9(2), 147-170. https://doi.org/10.1016/S1060-3743(00)00022-9

21. Self-, peer-, and teacher-assessments in Japanese university EFL writing classrooms - Sumie Matsuno, 2009. (n.d.). Retrieved July 1, 2019, from https://journals.sagepub.com/doi/abs/10.1177/0265532208097337

22. Gehringer, E. F., Chinn, D. D., PÃlrez-QuiÃśones, M. A., Ardis, M. A., Gehringer, E. F., Chinn, D. D., ... Ardis, M. A. (2005). Using peer review in teaching computing. ACM SIGCSE Bulletin, 37(1), 321-322. https://doi.org/10.1145/1047344.1047455

23. Sitthiworachart, J., & Joy, M. S. (2004). Effective peer assessment for learning computer programming.

24. Smith, J., Tessler, J., Kramer, E., & Lin, C. (2012). Using Peer Review to Teach Software Testing. Proceedings of the Ninth Annual International Conference on International Computing Education Research, 93-98. https://doi.org/10.1145/2361276.2361295

25. Turner, S., Pérez-Quiñones, M. A., Edwards, S., & Chase, J. (2011). Student Attitudes and Motivation for Peer Review in CS2. Proceedings of the 42Nd ACM Technical Symposium on Computer Science Education, 347-352. https://doi.org/10.1145/1953163.1953268

26. Tseng, S.-C., & Tsai, C.-C. (2007). On-line peer assessment and the role of the peer feedback: A study of high school computer course. Computers & Education, 49(4), 1161-1174. https://doi.org/10.1016/j.compedu.2006.01.007

27. Reily, K., Finnerty, P. L., & Terveen, L. (2009). Two Peers Are Better Than One: Aggregating Peer Reviews for Computing Assignments is Surprisingly Accurate. Proceedings of the ACM 2009 International Conference on Supporting Group Work, 115-124. https://doi.org/10.1145/1531674.1531692

28. Freeman, S., Eddy, S. L., McDonough, M., Smith, M. K., Okoroafor, N., Jordt, H., & Wenderoth, M. P. (2014). Active learning increases student performance in science, engineering, and mathematics. Proceedings of the National Academy of Sciences, 111(23), 8410-8415. https://doi.org/10.1073/pnas.1319030111

29. Yang, Y.-F. (2011). A reciprocal peer review system to support college students' writing.(Report). British Journal of Educational Technology, 42(4), 687-700. https://doi.org/10.1111/j.1467-8535.2010.01059.x

30. Cho, K., & Schunn, C. D. (2007). Scaffolded writing and rewriting in the discipline: A web-based reciprocal peer review system. Computers & Education, 48(3), 409-426. https://doi.org/10.1016/j.compedu.2005.02.004

31. Politz, J. G., Krishnamurthi, S., & Fisler, K. (2014). CaptainTeach: A Platform for In-flow Peer Review of Programming Assignments. Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education, 332-332. https://doi.org/10.1145/2591708.2602687

32. Gehringer, E. F. (2001). Electronic Peer Review and Peer Grading in Computer-science Courses. Proceedings of the Thirty-Second SIGCSE Technical Symposium on Computer Science Education, 139-143. https://doi.org/10.1145/364447.364564

33. Hyyrynen, V., HÃďmÃďlÃďinen, H., Ikonen, J., & Porras, J. (2010). MyPeerReview: An Online Peer-reviewing System for Programming Courses. Proceedings of the 10th Koli Calling International Conference on Computing Education Research, 94-99. https://doi.org/10.1145/1930464.1930481

34. de Raadt, M., Lai, D., & Watson, R. (2007). An Evaluation of Electronic Individual Peer Assessment in an Introductory Programming Course. Proceedings of the Seventh Baltic Sea Conference on Computing Education Research - Volume 88, 53-64. Retrieved from http://dl.acm.org/citation.cfm?id=2449323.2449330

35. Christian Schunn: Web-Based Peer Review and Learning. (n.d.). Retrieved July 5, 2019, from http://www.lrdc.pitt.edu/schunn/webpeers.html

36. Peerceptiv - Research Validated Peer Assessment. (n.d.). Retrieved July 5, 2019, from Peerceptiv website: https://www.peerceptiv.com/

APPENDICES

.1 APPENDIX Peerceptiv Grading

Peerceptiv Grading

There are 3 components of the Overall grade in Peerceptiv:

- a Writing grade – the grade received on the assignment submission by the peers and by the instructor (if the instructor wishes to participate in the assessment.) These may be written submissions or any assignment upload requiring formative feedback.
- a Reviewing grade – a measure of the accuracy of review ratings and helpfulness of review comments.
- a Task grade – a completion grade for all reviewing and back-evaluation tasks.

The Writing grade is an average of how reviewers rated the work product. First, the system calculates the reviewing accuracy of each of the reviewers (see above). Rather than an equally-weighted average across all reviewers, the ratings provided by reviewers with higher accuracy scores receive a greater weight. This prevents a student from being penalized by a random outlier rating. If an instructor or TA also grades a draft, those ratings are included in the Writing grade to what extent determined by the instructor in assignment setup.

The curve set during assignment setup (mean and standard deviation settings) is applied on Peerceptiv Writing and Reviewing grade components, so what matters is how students do relative to their classmates. When curved grading is selected for the Writing Grade, weighted peer ratings are used to rank the documents and grades are distributed according to the curve mean and standard deviation selected by the instructor.

Benchmark Grading is an option that delivers the top 5 and the bottom 5 work products to the instructor after the Review phase, allowing the instructor to grade those work products on a 0-100 scale. After the instructor grades those 10 work products, all other Writing Grades are distributed in-between the instructor setpoints in accordance with the peer ratings. The curve mean and standard deviation settings still apply to the Review Grade in assignments using Benchmark Grading for the Writing Grade.

There are two components to a student's Review grade: *accuracy* and *helpfulness*.

- Accuracy grades measure how closely the ranking order of the ratings a reviewer provides on peer documents corresponds to the ranking order of peer rating averages for each rating prompt on those same documents. When students provide ratings that track with the relative order of mean ratings for those same reviewing dimensions on the same documents, they are rewarded with a higher accuracy score. [The system calculates the Spearman correlation across the number of papers reviewed (n) x the number of rating prompts (m).] If a reviewer gives the same rating to all papers on all dimensions (e.g., 7s across the board), this necessarily results in an accuracy score of 0. Negative correlations are theoretically possible (i.e., an opposite view from everyone else) but rarely occur. Accuracy scores (-1 to 1) are then curved to produce accuracy grades (0 to 100) using the grading curve settings.
- The helpfulness grade is the extent to which authors believed a reviewer's comments were helpful and specific, as awarded by the back-evaluation ratings (5 high to 1 low.) The first step is to normalize each author's ratings (some authors might be especially harsh or overly nice across the board), and the final step is to curve the Helpfulness scores according to the instructor's settings.

Finally, the Task grade is awarded at 100% if a student performs all of the reviewing and back evaluation tasks required. If an instructor awards bonus points for extra reviewing, these bonus points appear as part of the Task grade. If bonus reviews are allowable, the student may do as many bonus reviews as the number of required reviews.

Late penalties are applied to the Review and Writing grades according to the per day penalty settings setup by the instructor. Each grade component is weighted according to the values selected by the instructor during assignment setup. Usually instructors make Writing grades and Reviewing grades count more than Task grades (e.g., the default 40% Writing, 40% Reviewing, and 20% Task weighting).

.2 APPENDIX Pre-Peer Review Surveys

Default Question Block

Before you do your peer review part of the assignment, we would like you to reflect on how well you think you and your classmates are going to do on the assignment.

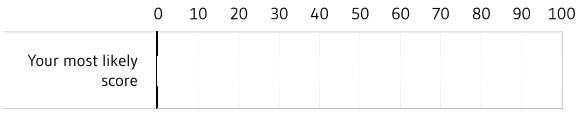
Please rate your likelihood in receiving a grade in the following ranges for this assignment:

F: < 60	0 %
D: 60 to 70	0 %
C: 70 to 80	0 %
B: 80 to 90	0 %
A: 90 to 100	0 %
Total	0 %

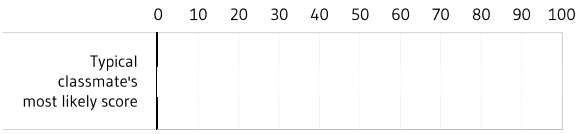
Please rate a typical, fall 2018, CS 160 student's likelihood in receiving a grade in the following ranges for this assignment.

F: < 60	0 %
D: 60 to 70	0 %
C: 70 to 80	0 %
B: 80 to 90	0 %
A: 90 to 100	0 %
Total	0 %

If you are to estimate the score you will most likely get on this assignment, what will it be? Provide a number between 0 and 100:



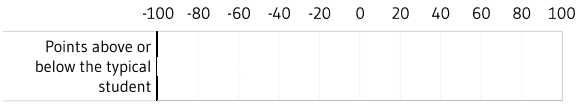
If you are to estimate the score the typical student will most likely get on this assignment, what will it be? Provide a number between 0 and 100:



What percentage of your classmates will have scores below yours on this assignment? (If you think your score will be among the best, choose 90% to 100%; If you think your score will be among the lowest, choose 0 to 10%; If you expect your score will be in the middle, choose 40% to 50% or 50% to 60%, depending on which percentile is closer to your expectation.)

- 0 to 10%
- 10% to 20%
- 20% to 30%
- 30% to 40%
- 40% to 50%
- 50% to 60%
- 60% to 70%
- 70% to 80%
- 80% to 90%
- 90% to 100%

How many points above or below the typical, fall 2018, CS 160 student do you think your score will be on this assignment? If you think your score will be above the typical student, slide to a positive number; if you think your score will be below the typical student, slide to a negative number:



.3 APPENDIX Post-Peer Review Surveys

Default Question Block

Now that you have finished your peer review, we would like you to reflect on your assignment, perception of peers, and the peer review activity.

How difficult was the assignment for you?

- ○ Extremely difficult
- ○ Somewhat difficult
- ○ Neither easy nor difficult
- ○ Somewhat easy
- ○ Extremely easy

How much have you thought about the critiques you received or tried to modify and improve your assignment after finishing the peer reviews in Peerceptiv? Please answer honestly as this will NOT be tied to your grades in any way.

- ○ A great deal
- ○ A lot
- ○ A moderate amount
- ○ A little
- ○ None at all

Overall, how useful did you find reviewing your peers' assignments to be?

- ○ Extremely useful
- ○ Very useful
- ○ Moderately useful
- ○ Slightly useful
- ○ Not at all useful

Overall, how useful did you find receiving reviews to be?

- ○ Extremely useful
- ○ Very useful
- ○ Moderately useful
- ○ Slightly useful
- ○ Not at all useful

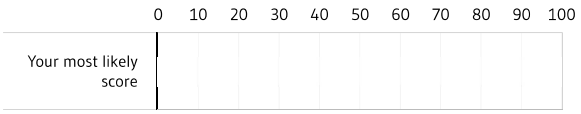
Please rate your likelihood in receiving a grade in the following ranges for this assignment (percentages must add up to 100).

F: < 60	0 %
D: 60 to 70	0 %
C:70 to 80	0 %
B: 80 to 90	0 %
A: 90 to 100	0 %
Total	0 %

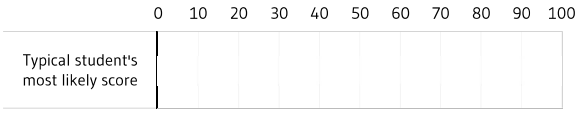
Please rate a typical, fall 2018, CS 160 student's likelihood in receiving a grade in the following ranges for this assignment (percentages must add up to 100).

F: < 60	0 %
D: 60 to 70	0 %
C:70 to 80	0 %
B: 80 to 90	0 %
A: 90 to 100	0 %
Total	0 %

If you are to estimate the score you will most likely get on this assignment, what will it be? Provide a number between 0 and 100:



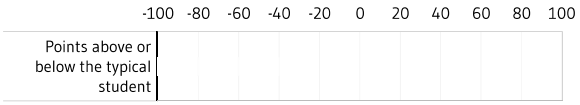
If you are to estimate the score the typical student will most likely get on this assignment, what will it be? Provide a number between 0 and 100:



What percentage of your classmates will have scores below yours on this assignment? (If you think your score will be among the best, choose 90% to 100%; If you think your score will be among the lowest, choose 0 to 10%; If you expect your score will be in the middle, choose 40% to 50% or 50% to 60%, depending on which percentile is closer to your expectation.)

- ○ 0 to 10%
- ○ 10% to 20%
- ○ 20% to 30%
- ○ 30% to 40%
- ○ 40% to 50%
- ○ 50% to 60%
- ○ 60% to 70%
- ○ 70% to 80%
- ○ 80% to 90%
- ○ 90% to 100%

How many points above or below the typical, fall 2018, CS 160 student do you think your score will be on this assignment? If you think your score will be above the typical student, slide to a positive number; if you think your score will be below the typical student, slide to a negative number:



.4 APPENDIX Pre-Survey

Consent

WHAT IS THE PURPOSE OF THIS FORM?
This form contains information you will need to help you decide whether to be in this research study or not. Please read the form carefully and ask the study team member(s) questions about anything that is not clear.
WHY IS THIS RESEARCH STUDY BEING DONE?
The purpose of this research study is to determine the effects of teaching CS 160, Computer Science Orientation, using various pedagogical strategies and different computer programming languages. We want to investigate the correlation between specific teaching methods and programming languages in CS 160 with drop, failure, withdraw (DWF) rates, gender, ethnicity, race, class standing, declared major, knowledge of specific computer science concepts, details within assignments and labs, feelings about learning a specific programming language or the use of a book in the course, and grades in CS 160, as well as in the two subsequent courses CS 161 and 162. Some findings of the study will be used for students' thesis/dissertation. The study team members include Dr. Jennifer Parham-Mocello, Fengfei Zheng, Asma Alghamdi, Eman Almadhoun, and Christopher Kawell.
WHY AM I BEING INVITED TO TAKE PART IN THIS STUDY?
You are being asked to take part in this study because you are currently a student in a section of CS 160 or you are in CS 161 or CS 162 and took CS 160 in the Fall 2017, 2018, or 2019 at Oregon State University.
What will happen if I take part in this research study?
The study activities include you participating in the course as you normally would conduct yourself. There is nothing extra that you need to do that isn't already required by the course or subsequent courses. Your grades (all associated with these courses), submitted assignments, and pre/post surveys will be collected for analysis. We will also obtain the following information from the Office of the Registrar: DWF rates, grade distributions, gender, ethnicity, race, declared major, class standing. You will be asked to complete a 10-minute survey about your labs in this course. All identifiable information will be wiped from this data and replaced with randomized IDs.All identifiable information will be wiped from this data and replaced with randomized IDs.
Study duration: the length of time you continue in CS 160, CS 161, and CS 162. Your decision to take part or not take part in this study will not affect your grades, your relationship with your professors, or standing in the University.
Use of Data: Because it is not possible for us to know what studies may be a part of our future work, we ask that you give permission now for us to use your personal information without being contacted about each future study. Future use of your information will be limited to studies about computer science education, in particular the correlation between teaching methods and programming languages with any associated material in other cs courses or in the degree, such as pro-school acceptance, change of major, gender, ethnicity, race, class standing, declared major, etc. If you agree now to future use of your personal information but decide in the future that you would like to have your personal information removed from the research database, please contact Jennifer Parham-Mocello at parhammj@eecs.orst.edu prior to Summer 2019. Once we destroy the identifiers, we will be unable to remove your data from the larger data set.

- ○ You may store my information for use in future studies.
- ○ You may not store my information for use in future studies.

WHAT ARE THE RISKS AND POSSIBLE DISCOMFORTS OF THIS STUDY?
There are minimal risks involved in participating in this study. Since there will be identifiers on the course information, the research team will do its best to keep data secure and confidential. There are no major risks involved in participating in this study. While the research team will keep the participants' course and information and survey responses confidential, there is always a risk that they could accidentally disclose information that identifies participants. Furthermore, both the security and confidentiality of information collected online through Canvas and Qualtrics cannot be guaranteed. Information collected online or sent by email can be intercepted, corrupted, lost, destroyed, arrive late or incomplete, or contain viruses. The research team will do its best to keep data secure and confidential.
WHAT ARE THE BENEFITS OF THIS STUDY?
This study is not designed to benefit you directly although as a byproduct of research you will help advance future CS 160 courses creating a better orientation class for OSU. This will help determine better instruction for the course helping future students and possibly yourself, if you receive the method of teaching or programming language proving to be the most effective. In addition, this study will help to determine a better use of limited classroom resources for large classes.
WILL I BE PAID FOR BEING IN THIS STUDY?
You will not be paid for being in this research study.
WHO WILL SEE THE INFORMATION I GIVE?
The information you provide during this research study will be kept confidential to the extent permitted by law. Research records will be stored securely and only researchers will have access to the records. Federal regulatory agencies and the Oregon State University Institutional Review Board (a committee that reviews and approves research studies) may inspect and copy records pertaining to this research. Some of these records could contain information that personally identifies you. If the results of this project are published, your identity will not be made public. The results will be presented to EECS faculty and submitted to Special Interest Group in Computer Science Education and other CS education conferences and journals.
What other choices do I have if I do not take part in this study?
Participation in this study is voluntary. If you decide to participate, you are free to withdraw at any time without penalty. If you choose to withdraw from this project before it ends, the researchers may keep information collected about you and this information may be included in study reports. Your decision to take part or not take part in this study will not affect your grades, your relationship with your professors, or standing in the University.
WHO DO I CONTACT IF I HAVE QUESTIONS?
If you have any questions about this research project, please contact: Jennifer Parham-Mocello at parhammj@eecs.orst.edu. If you have questions about your rights or welfare as a participant, please contact the Oregon State University Institutional Review Board (IRB) Office, at (541) 737-8008 or by email at IRB@oregonstate.edu.
WHAT DOES MY AGREEMENT ON THIS CONSENT FORM MEAN?
Your agreement indicates that this study has been explained to you, that your questions have been answered, and that you agree to take part in this study.

- ○ I agree to participate in this study.
- ○ I do not agree to participate in this study.

Participant Info

Course Section:

- ○ 010: 2-2:50pm
- ○ 020: 8-8:50am
- ○ 030: 10-10:50am
- ○ Honors: 9-9:50am

Why did you register for this section of CS 160?

- ○ It was the last section available
- ○ I was interested in the section topic
- ○ It was a time that fit my schedule
- ○ I wanted to take an honors course
- ○ Other

What other reason led you to register for the specific section of CS 160?

Please rate your interest level in the following.

	Extremely Interested	Somewhat Interested	Not Interested At All
This class.	○	○	○
Learning more about computer science.	○	○	○
Learning more about programming/coding.	○	○	○
Majoring in computer science.	○	○	○
Taking more computer science classes.	○	○	○
Using computation in my job after college.	○	○	○

Prior Programming Languages

How much programming experience (in any language) do you have before taking CS 160? (Choose the closest that applies)

- ○ None
- ○ less than 6 months
- ○ 6 months - 1 year
- ○ 1 - 2 years
- ○ 2 - 3 years
- ○ 3 - 4 years
- ○ more than 4 years

Which of the following languages have you programmed in before? (Select all that apply)

- ☐ Block-based (Scratch, Snap, Alice)
- ☐ Imperative/OOP (C,C++,C#, Python, Java)
- ☐ Functional (Scheme/Racket, Lisp, Haskell, OCaml)
- ☐ Scripting Language (JavaScript, PHP, NodeJS)
- ☐ Mathematical Language (MATLAB, Maple)

Do you think programming is something that people are naturally good/poor at doing or is something that people can improve with practice?

- ○ naturally good/poor
- ○ skill improves with practice
- ○ combination of naturally good and improve with practice

Were you self-taught or did you take classes in programming?

- ○ Self-taught
- ○ Took classes

How much experience do you have using the Python programming language?

- ○ less than 6 months
- ○ 6 months - 1 year
- ○ 1 - 2 years
- ○ 2 - 3 years
- ○ 3 - 4 years
- ○ more than 4 years

Comparative Judgement

On a scale of 5, rate your level of competency in the following:

	Novice 1	2	3	4	Expert 5
programming/coding					
solving problems					
computer science knowledge					
Python programming					

On a scale of 5 , rate your perception of the majority of your classmates' level of competency in the following:

	Novice 1	2	3	4	Expert 5
programming/coding					
solving problems					
computer science knowledge					
Python programming					

Relative to yourself, where would you rate the majority of your classmates in CS160 for the following?

	a great deal above me	a little above me	at the same level	a little below me	a great deal below me
programming/coding	○	○	○	○	○
solving problems	○	○	○	○	○
computer science knowledge	○	○	○	○	○
Python programming	○	○	○	○	○

Relative to your classmates in CS160, where would you rate yourself for the following?

	top 20%	top 40%	average	bottom 40%	bottom 20%
programming/coding	○	○	○	○	○
solving problems	○	○	○	○	○
computer science knowledge	○	○	○	○	○
Python programming	○	○	○	○	○

Robotics/Lab Info

How would the following affect your **interests in this class**?

	Greatly Increase	Slightly Increase	It wouldn't	Slightly Decrease	Greatly Decrease
Using robots in the labs	○	○	○	○	○
Writing programs/code	○	○	○	○	○
Combining the labs with other ENGR majors	○	○	○	○	○
Using 3D Printing	○	○	○	○	○
Using stories to explain computation	○	○	○	○	○
Working in teams	○	○	○	○	○
Peer review of assignments	○	○	○	○	○

How would the following affect your **motivation to learn more about CS**?

	Greatly Increase	Slightly Increase	It wouldn't	Slightly Decrease	Greatly Decrease
Using robots in the labs	○	○	○	○	○
Writing programs/code	○	○	○	○	○
Combining the labs with other ENGR majors	○	○	○	○	○
Using 3D Printing	○	○	○	○	○
Using stories to explain computation	○	○	○	○	○
Working in teams	○	○	○	○	○
Peer review of assignments	○	○	○	○	○

How would the following affect your **motivation to learn more about programming/coding**?

	Greatly Increase	Slightly Increase	It wouldn't	Slightly Decrease	Greatly Decrease
Using robots in the labs	○	○	○	○	○
Writing programs/code	○	○	○	○	○
Combining the labs with other ENGR majors	○	○	○	○	○
Using 3D Printing	○	○	○	○	○
Using stories to explain computation	○	○	○	○	○
Working in teams	○	○	○	○	○
Peer review of assignments	○	○	○	○	○

What size team would you prefer to work with in the following environments?

	Individually	2 people	3 people	4 people	more than 4 people
Labs	○	○	○	○	○
Assignments	○	○	○	○	○
Class Exercises	○	○	○	○	○
Term Project	○	○	○	○	○

Your lab and project teams will have students from MIME 101, the Mechanical, Industrial, and Manufacturing Engineering orientation course. Do you think a more diverse group of majors will add to your ability to learn more in this class?

- ○ Yes
- ○ Maybe
- ○ No

Have you ever ...

	Yes	No
Used a Cozmo robot	○	○
Used a Lego Mindstorm robot	○	○
Designed anything for 3D printing	○	○

Algorithms

An algorithm ...

	What is your answer?			How confident are you in your answer?			
	Always	Sometimes	Never	Absolutely Certain	Pretty Sure	Informed Guess	No Idea
... is given in a language.	○	○	○	○	○	○	○
... solves a problem.	○	○	○	○	○	○	○
... terminates.	○	○	○	○	○	○	○
... takes time to execute.	○	○	○	○	○	○	○

What are the properties of an algorithm?

	What is your answer?		How confident are you in your answer?			
	must	should	Absolutely Certain	Pretty Sure	Informed Guess	No Idea
All instructions in an algorithm _____ be effective.	○	○	○	○	○	○
An algorithm _____ have good runtime complexity.	○	○	○	○	○	○

How many algorithms are there for a problem?

	What is your answer?			How confident are you in your answer?			
	exactly 1	1 or more	0 or more	Absolutely Certain	Pretty Sure	Informed Guess	No Idea
For every problem there is/are _____ algorithms to solve it.	○	○	○	○	○	○	○

How is the efficiency of an algorithm measured?

	What is your answer?			How confident are you in your answer?			
	in seconds	in number of steps	as a growth rate	Absolutely Certain	Pretty Sure	Informed Guess	No Idea
The efficiency of an algorithm is measured _____.	○	○	○	○	○	○	○

Representation

Representation

	What is your answer?		How confident are you in your answer?			
	True	False	Absolutely Certain	Pretty Sure	Informed Guess	No Idea
A data type defines how to store data.	○	○	○	○	○	○
Different data structures can implement one data type.	○	○	○	○	○	○
A data type can be used to implement a data structure.	○	○	○	○	○	○
Recursion is more general than loops.	○	○	○	○	○	○

Abstraction

Abstraction

	What is your answer?		How confident are you in your answer?			
	True	False	Absolutely Certain	Pretty Sure	Informed Guess	No Idea
Abstraction is the same as generalization.	○	○	○	○	○	○
Algorithms are abstractions.	○	○	○	○	○	○
Types are abstractions.	○	○	○	○	○	○
Algorithm runtime is an abstraction.	○	○	○	○	○	○